CS 378 Project 2, due Thursday, **09/19**.

Part 1: Goals

This project has three main goals:

- extract some structured data from your text, pdf or images with Gemini
- load your structured data files (csv, json) from GCS to BQ
- analyze your data collection against some common data anomalies types to satisfy our acceptance criteria

Part 2: Code Samples

The `project2` folder of the snippets repo has some code samples and other artifacts that are relevant for this project.

Note: You can skip the two extraction notebooks (`1-air-travel-extract-*.ipynb`). However, you should run the data load notebook (`2-air-travel-load.ipynb`) in your own project. This will load the air travel data to your BQ instance and help you follow along in future projects.

Part 3: Acceptance Criteria

Your data collection needs to meet certain criteria for subsequent projects to make sense and for them to have enough scope. The criteria are formulated as a series of data anomaly types, which you will check your data against. You want your data collection as a whole to suffer from all 10 anomaly types. This means that you have sufficiently messy data to work with, which is what we are aiming for.

Please review the list of criteria below and evaluate which elements of your data collection satisfy each one. Note that criteria 5-10 represent different anomaly types that should be present in the data.

Criteria	Description	Applicable to Project	Air Travel Examples
1	Warehouse must be made up of multiple independent data sources. You need at least 4 data sources.	All	Airport Guide, Open Flights, BTS, TSA, etc.
2	Warehouse must be composed of at least one source whose type is unstructured. This can be text, pdf, or images.	All	Airport Businesses and TSA Traffic were both created from pdf files.

3	Warehouse must be composed of multiple logical entities.	All	Airports, Airlines, Airport Businesses, Airport Reviews, Flights, Routes, Countries, Aircraft
4	Functional dependencies should hold across all tables such that the values within a record are consistent.	All	The name of an airport, its city, state, country and code need to make sense. This is mostly to guard against synthetic data that is randomly generated.
5	There exists a field in any table of the warehouse whose assigned data type does not best fit its domain of values.	3	airports.timezone stores a numeric value as a string. tsa_traffic.date is stored as a string instead of date
6	There exists a field in any table of the warehouse whose null values are represented as empty strings, "\n" or something similar.	3	source_airport_id in the flight_routes table
7	There exists a field in any table of the warehouse that stores the values of multiple attributes in a single cell. The values represent different attributes.	3	flight_delays.airport_name is composed of city, state, and airport. All three attributes are stored in the same column
8	There exists a field in any table of the warehouse that stores multiple values in the same cell. The values represent a list of elements for the same attribute.	4	flight_routes sometimes stores a list of equipments in the same cell, airport_businesses sometimes stores a list of menu items in the same cell.
9	There exists two tables in the warehouse which originated from different sources and which have similar data. Moreover, the tables in question use two different identifier systems to refer to the same entity.	4	airport information is repeated across multiple tables in a non-standard way. See for example airportRef and aiportIdent versus airport_id and airport_code.
10	There exists a table in the warehouse that models more than one logical entity. This can lead to storing repeated values within the same table.	4	The flight_delays table has information about airports, airlines, and flight delays. Fields like carrier_name

			and airport_name shouldn't be in this table.

What should you do if your warehouse is missing one or more criteria?

- Think about ways to broaden your data domain and start looking for additional related datasets.
- Consider choosing a different domain that has more available data. If you go down this route, please note that you'll need to redo most of Project 1 in a relatively short timespan.
- Given that you have two weeks to complete this assignment, I expect your data to satisfy all 10 criteria, including the six anomaly types. If you are missing one or more criteria, please be sure to get sign off from the Professor.

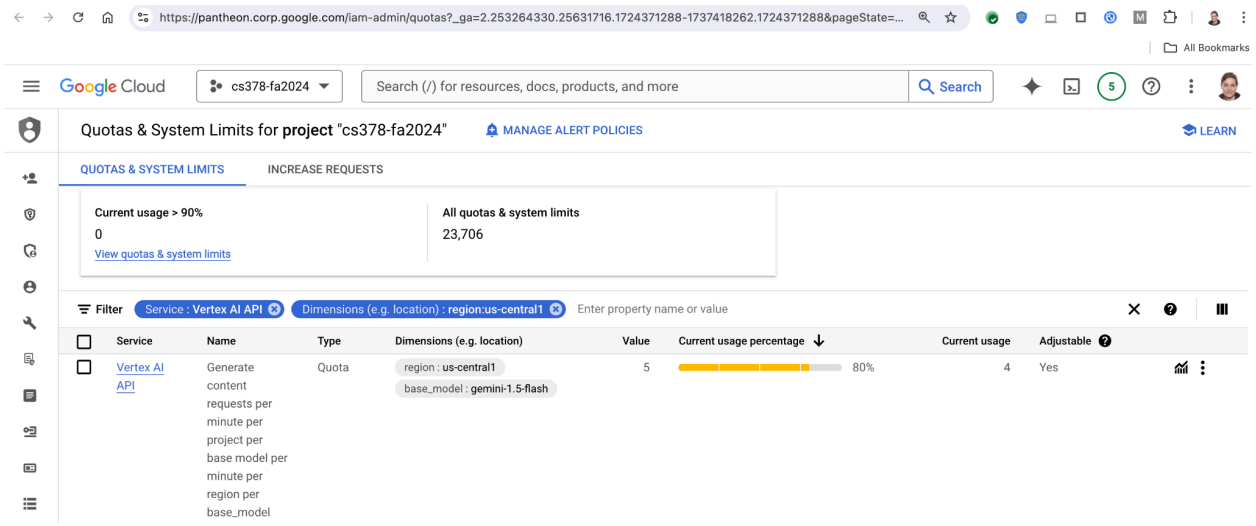
Part 4: Implementation Guidelines

- Create a new folder in your repo and name it `project2`. Store all of your artifacts for this project in the `project2` folder.
- Develop a Colab notebook that extracts some interesting data from your unstructured dataset and save the results as csv or json files stored in your bucket on GCS. Name your notebook `1-[your-domain]-extract-[your-dataset].ipynb`.
- Develop a Colab notebook that loads the data files into BQ. Load each file into its own table in the raw area. The only exception is if you have a collection of files which represent the same type of data and are split into multiple files by date. In that case, you want to load all the files into the same table. Name this notebook `2-[your-domain]-load.ipynb`.
- Annotate your notebooks with section headers and short Markdown comments to improve their readability.
- Store your BQ tables in a raw dataset. The dataset name should follow the naming convention of `[your-domain]_raw`.
- When creating the BQ tables, add two new columns to the end of each table as follows:
  - `_data_source (STRING)`: should default to the name of the data source. Choose a descriptive name to identify each data source (e.g. "openflights").
  - `_load_time (TIMESTAMP)`: should default to the current timestamp and represent the time in which the records were loaded into the table.
- Choose descriptive table names. Note that the name of your table can be different from the name of the file from which it is sourced.
- Lowercase both the table and column names. Use underscores (instead of hyphens or camel case) to name a table name or column name with multiple words in its name.
- Update your data dictionary from Project 1 with the work that you've done in this project. For example, update the attribute list and any other details which have changed.
- Update your ERD with the work you've done in this project. You do not need to add the `_data_source` and `_load_time` fields to the diagram as those fields are understood and would only end up cluttering the diagram.

- Review the acceptance criteria in Part 3 and cross-reference the list of anomaly types against your data collection. Document how your data satisfies each anomaly using a specific example. If your data is missing an anomaly or if you're unsure if the anomaly applies to it, make note of that as well and speak to the Professor or TA. Name your document `anomaly-analysis.md`.
- Publish to your repo: `1-[your-domain]-extract.ipynb`, `2-[your-domain]-load.ipynb`, `[your-domain]-data-dict.xlsx`, and `[your-domain]-erd.pdf`, and `anomaly-analysis.md`. Remember that all artifacts should go into your `project2` folder.
- Create a submission.json file and upload it to Canvas by the deadline. Only one person per group needs to do this step.

Part 5: Implementation Hints

- When you start working with Gemini, you will probably hit a quota limit. The quota is set ridiculously low by default (5 requests / minute). You can apply for a quota increase from here. You should ask for 200 requests / minute. It should get approved automatically within 5 minutes unless you are on the free trial. If on the free trial, you should switch billing accounts before requesting the quota increase. Please speak to the Professor if you face any issues.



- When loading the data into BQ, you may need to relax the table schema constraints if you run into problems. For example, if you have defined a field as mandatory, you may need to redefine it as nullable. If you have defined a field as a DATE type, you may need to redefine it as a STRING. The goal here is to get the data into BQ and massage it later. This is known as an "ELT" approach (as opposed to "ETL").

Grading Rubric

Due Date: 09/19/24

`1-[your-domain]-extract.ipynb` correctly written and indicates successful extraction **-5** for each missing output **-10** did not extract structured data with LLM **-10** did not store output from extraction in csv or json **-15** unable to verify the output from extraction process with GCS commands **-30** missing file **-20** Didn't run the code	30
`2-[your-domain]-load.ipynb` correctly written and indicates successful table creation **-5** for each missing output **-5** column names and/or table names not lowercase **-5** did not include data_source or load_time fields in tables **-5** did not update variable values (project_id, bucket_name, etc) **-15** unable to verify loads through bigquery commands **-30** missing file **-20** Didn't run the code	30
`anomaly-analysis.md` is thorough and has all information. All 10 criteria, including the six anomaly types should be met unless you have received a sign off from the Professor. **-5** for each criteria explanation not thorough **-5** if crucial misunderstanding of each criteria **-30** missing file	30
`[your-domain]-data-dict.xlsx` and `[your-domain]-erd.pdf` **-5** for each missing important link in ERD **-5** data dictionary does not contain essential information of a table **-5** ERD not aligned with data dictionary columns **-10** missing ERD or data dictionary	10

`submission.json` submitted into Canvas. Your project **will not** be graded without this submission. The file should have the following schema:
 	
`{`
`  "commit-id": "your most recent commit ID from Github",`
`  "project-id": "your project ID from GCP"`
`}`
 	
Example:
 	
`{`
`  "commit-id": "dab96492ac7d906368ac9c7a17cb0dbd670923d9",`
`  "project-id": "some-project-id"`
`}`
	Required

Total Credit:	**100**